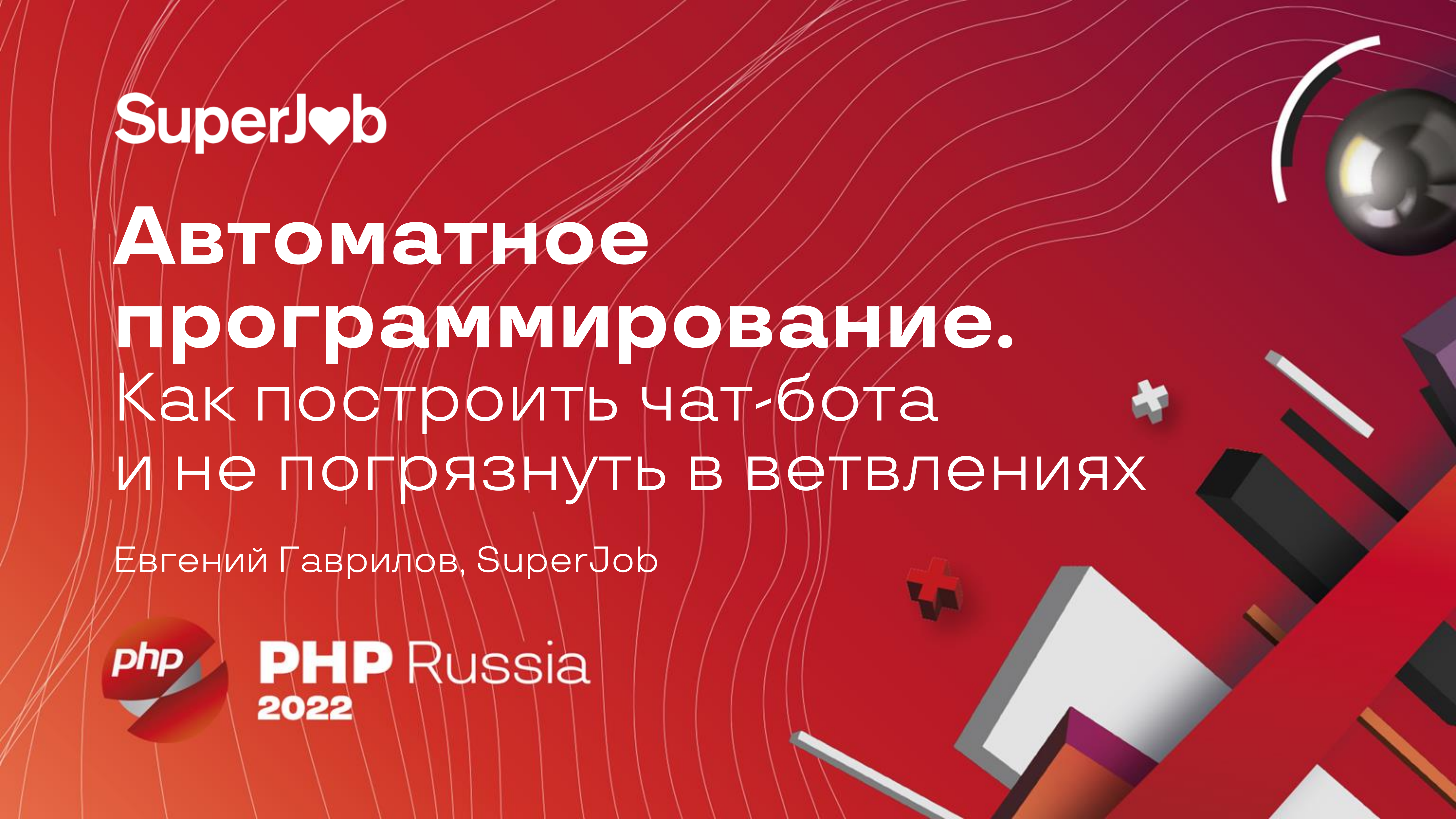

SuperJob
Автоматное программирование.
Как построить чат-бота
и не погрязнуть в ветвлениях
Евгений Гаврилов, SuperJob
php
PHP Russia
2022

Общение в мессенджере

Добрый день, Финн. На связи **ООО ЗВЕЗДА**.
У нас есть вакансия **Оператор звезды**.

Мы предлагаем:
- комфортабельный офис с видом на планеты
- гибкий график
- полный соц. пакет
- а ещё у нас есть **печеньки**

Скажите, вам интересна вакансия?

Да

Спасибо, с вами обязательно свяжутся





Работа с готовым решением

SuperJob

Результат

Данные

ПК менеджера

Результат

Данные





Недостатки готового решения

- Отсутствует кастомизация сценария
- Много ручной работы
- Отсутствует API для интеграции





Альтернативы

- Сторонние сервисы
- Собственная разработка

Сторонние сервисы

Плюсы:

- No-code-редакторы
- API
- Зоопарк интеграций





Сторонние сервисы

Минусы:

- Стоимость интеграции
- Vendor lock-in

Собственная разработка

Плюсы:

- Делаем всё, что хотим
- Делаем как захотим

Собственная разработка

Минусы:

- Нужны компетенции
- Без компетенций дорого и долго





Работа с мессенджером

Получение сообщения

Обработка

Отправка сообщения





Классическая разработка

Плюсы:

- Быстро
- Дешево
- Счастливый бизнес





Классическая разработка

Минусы:

- Потенциальные проблемы в будущем





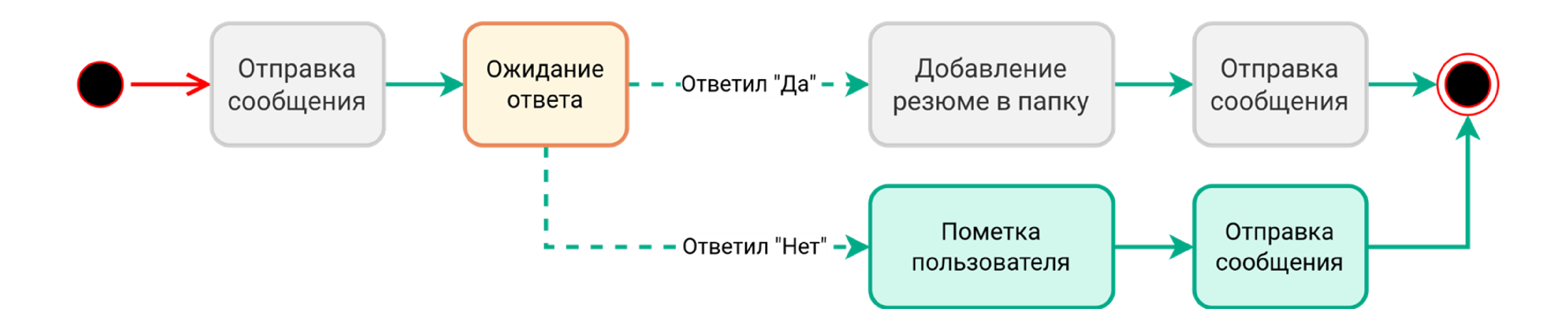
Итерация 1: легко



Итерация 1: легко

```
if ($message === 'Да') {
    // добавляем резюме в ЛК работодателя
    // отправляем сообщение
}
```

Итерация 2: небольшое ветвление







Итерация 2: небольшое ветвление

```
if ($message === 'Да') {
    // ...
} else {
    // помечаем пользователя
    // отправляем сообщение
}
```





Итерация 3: новый флаг

Отправка сообщения → Ожидание ответа

Ожидание ответа — Ответил "Да" → Добавление резюме в папку → Отправка сообщения → конец

Ожидание ответа — Ответил "Нет" → Пометка пользователя → Отправка сообщения → конец

Ожидание ответа — Нет ответа → Отправка сообщения → Ожидание ответа

Итерация 3: новый флаг

```
if ($start) {
  if ($message === 'Да') {
    // ...
  } else {
    // ...
  }
} else {
  // сохраняем причину отказа
  // отправляем сообщение
}
```





Итерация 4

Отправка сообщения
Ожидание ответа
Ответил "Да"
Добавление резюме в папку
Отправка сообщения
Ответил "Нет"
Пометка пользователя
Отправка сообщения
Нет ответа
Отправка сообщения
Отправка сообщения
Добавление причины отказа
Ожидание ответа
причина отказа

Итерация 4: ветки в ветках

```
if ($start) {
    if ($message === 'да') {
        // ...
    } elseif ($message === 'нет') {
        $start = false;
        // ...
    } else {
        // отправляем сообщение
    }
} else {
    // ...
}
```





Итерация 5...



```
if (...) {
    if (...) {
        // ...
    } elseif (...) {
        // ...
        if (...) {
            // ...
        } else {
            // ...
        }
    } else {
        // ...
    }
} else {
    // ...
}
```

Итерация 5...6...



```
if (...) {
    if (...) {
        // ...
    } elseif (...) {
        // ...
        if (...) {
            // ...
        } else {
            // ...
        }
    } else {
        // ...
    }
} else {
    // ...
}
```

```
if (...) {
    if (...) {
        // ...
    } elseif (...) {
        // ...
        if (...) {
            // ...
            if (...) {
                // ...
            } else {
                // ...
            } elseif(...) {
                // ...
            }
        } else {
            // ...
        }
    } else {
        // ...
    }
} else {
    // ...
}
```





Итерация 5…6…7…



```
if (...) {
    if (...) {
        // ...
    } elseif (...) {
        // ...
        if (...) {
            // ...
        } else {
            // ...
        }
    } else {
        // ...
    }
} else {
    // ...
}
```

```
if (...) {
    if (...) {
        // ...
    } elseif (...) {
        // ...
        if (...) {
            // ...
            if (...) {
                // ...
            } else {
                // ...
            } elseif(...) {
                // ...
            }
        } else {
            // ...
        }
    } else {
        // ...
    }
} else {
    // ...
}
```

```
if (...) {
    if (...) {
        if (...) {
            // ...
        } else {
            // ...
        }
    } elseif (...) {
        // ...
        if (...) {
            // ...
            if (...) {
                if (...) {
                    // ...
                } else {
                    // ...
                }
            } else {
                // ...
            } elseif(...) {
                if (...) {
                    // ...
                    if (...) {
                        // ...
                    } else {
                        // ...
                    }
                } else {
                    // ...
                }
            }
        } else {
            // ...
        }
    } else {
        // ...
    }
} else {
    // ...
}
```

Итерация 7...20...

Прошло полгода...



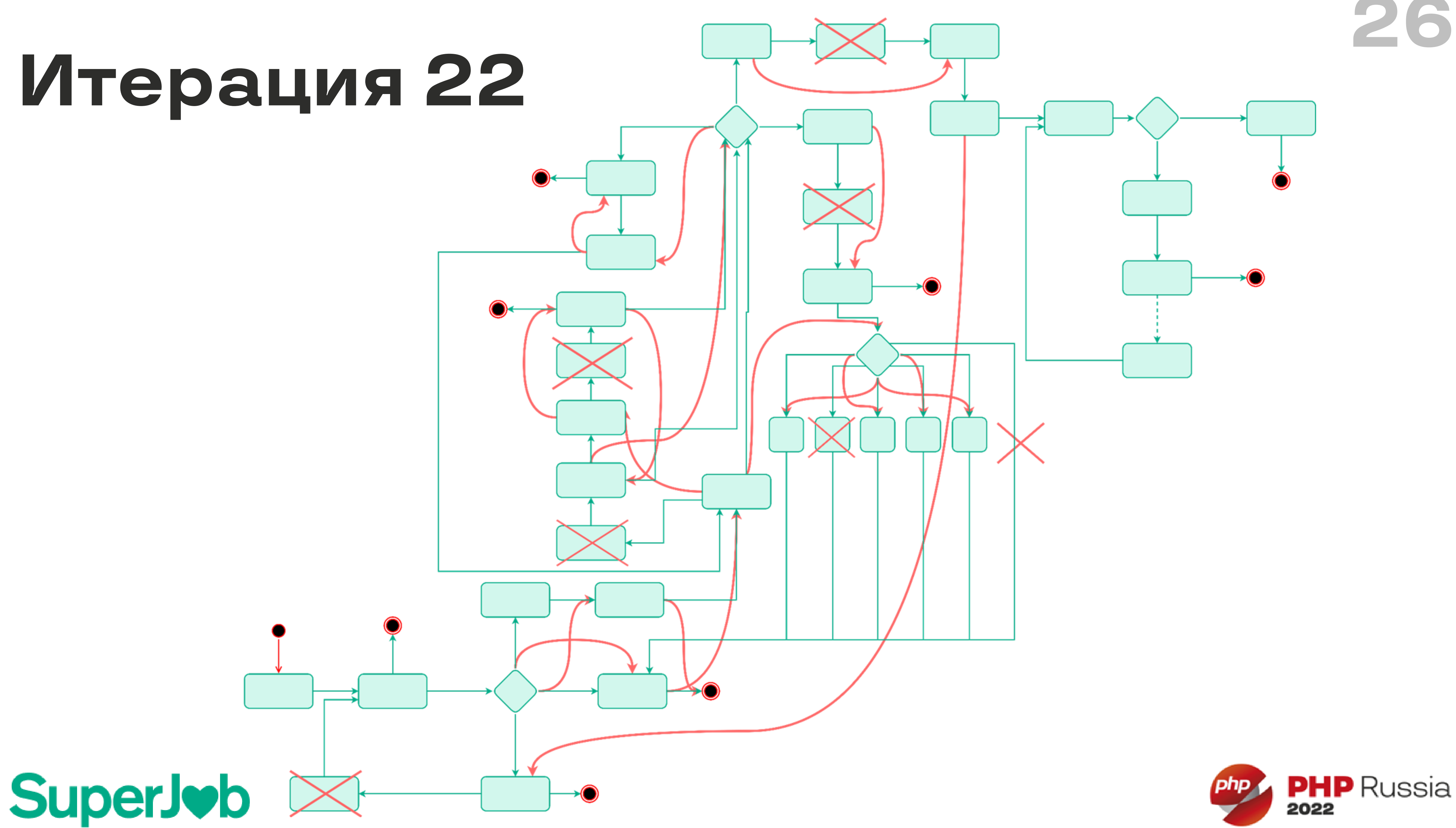


Итерация 21

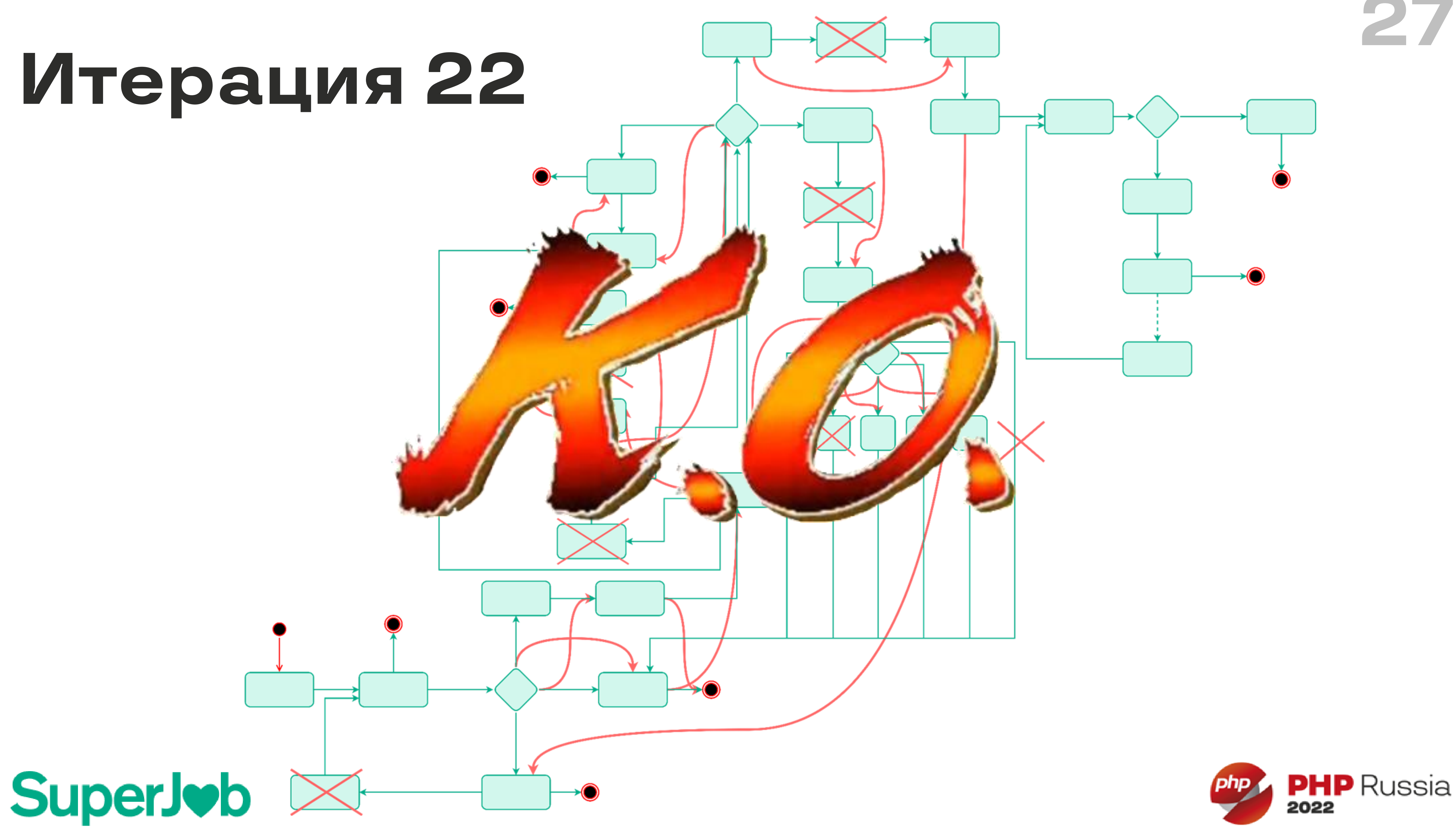
SuperJob



Итерация 22



Итерация 22



Классическая разработка

Минусы:

- Отсутствует целостная картина поведения





Классическая разработка

Минусы:

- Отсутствует целостная картина поведения
- Высокая сложность расширения и изменения





Классическая разработка

Минусы:

- Отсутствует целостная картина поведения
- Высокая сложность расширения и изменения
- Высокая вероятность ошибок





Классическая разработка

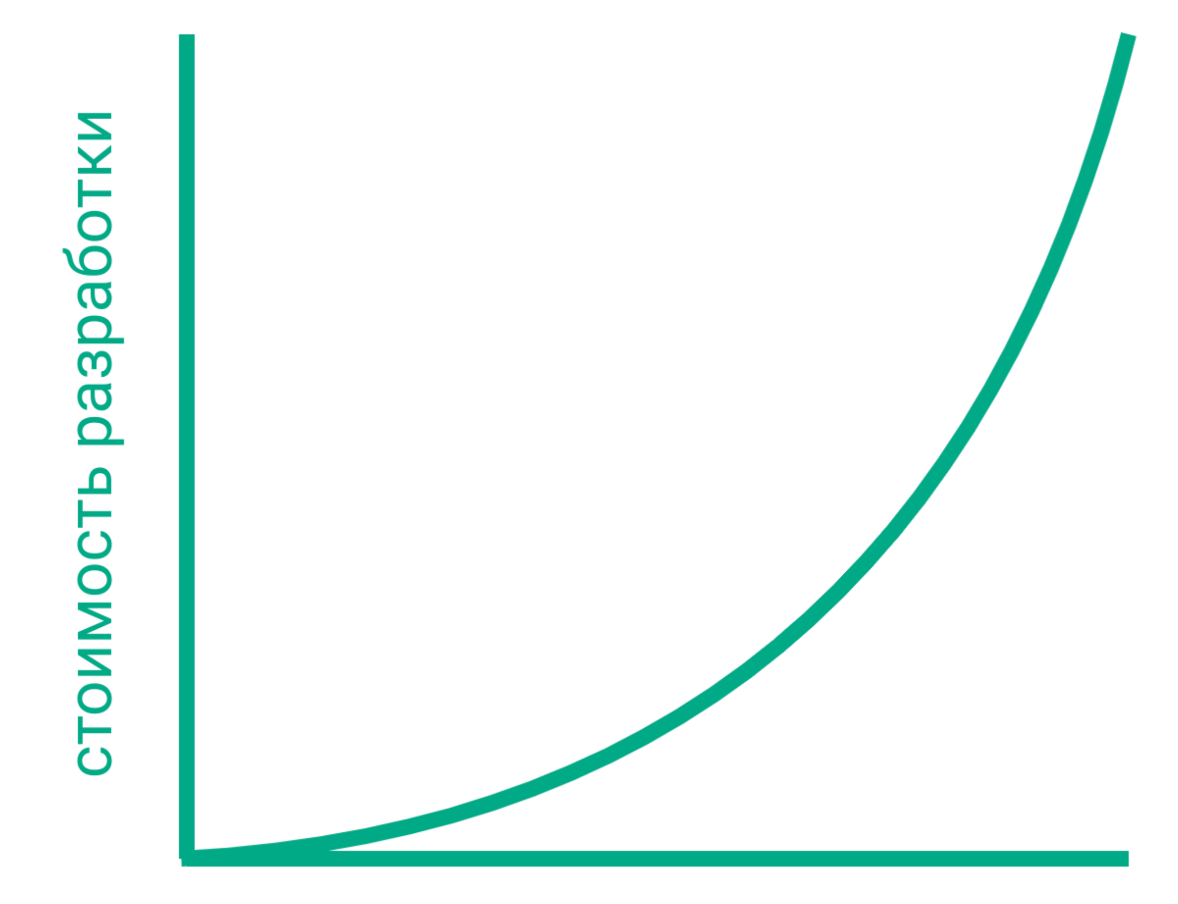
Минусы:

- Отсутствует целостная картина поведения
- Высокая сложность расширения и изменения
- Высокая вероятность ошибок
- Сложно тестировать





Дорогая разработка







Декомпозиция спасёт ботов!





Стадии декомпозиции бота.
Отрицание



Гаврилов Евгений 22:48

Я придумал! Просто распределю логику по компонентам.

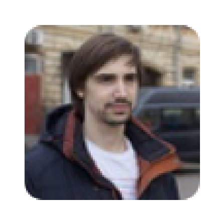






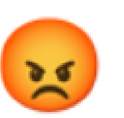


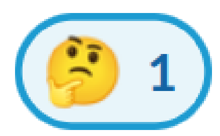
Стадии декомпозиции бота.
Гнев



Гаврилов Евгений 11:46

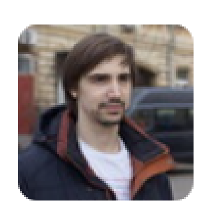
Что за бред??

Я не понимаю как сюда новую логику вкорячить 





Стадии декомпозиции бота. Торг



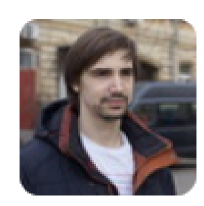
Гаврилов Евгений 23:06

Если костылями, то за 15 часов управлюсь.





Стадии декомпозиции бота. Депрессия



Гаврилов Евгений 23:07

Я уже 4 часа чиню тесты 😞

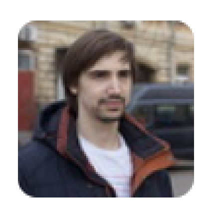






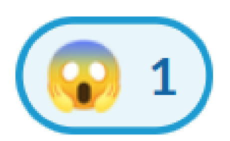


Стадии декомпозиции бота.
Принятие



Гаврилов Евгений 11:40

Похоже, тут надо всё переписать...





Декомпозиция НЕ спасёт ботов ;(





Автоматное программирование,

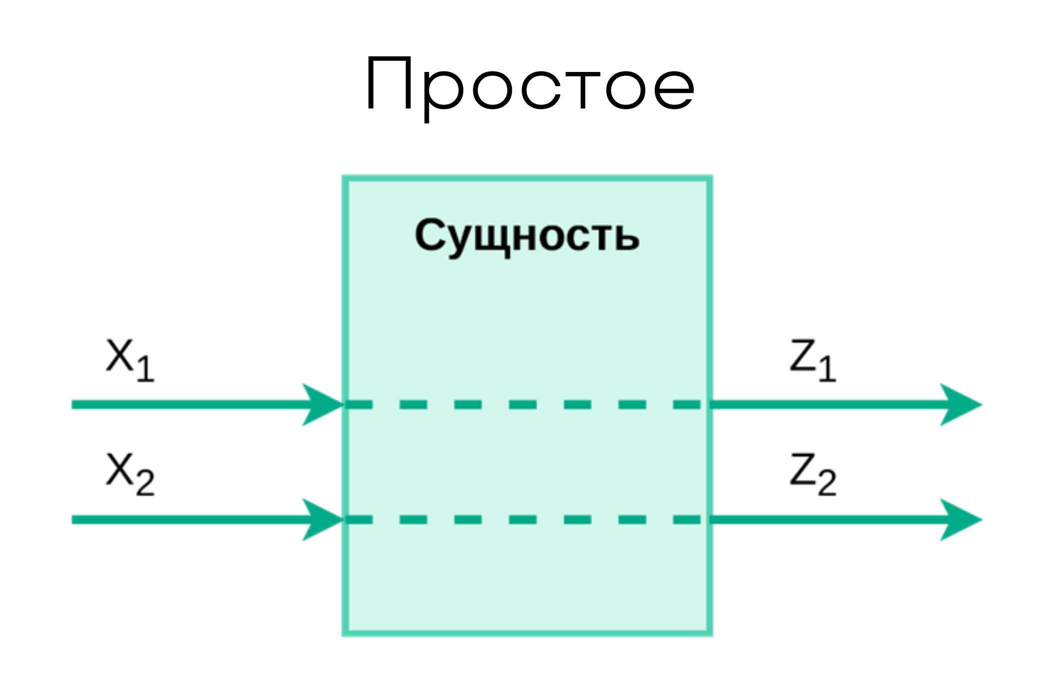
или

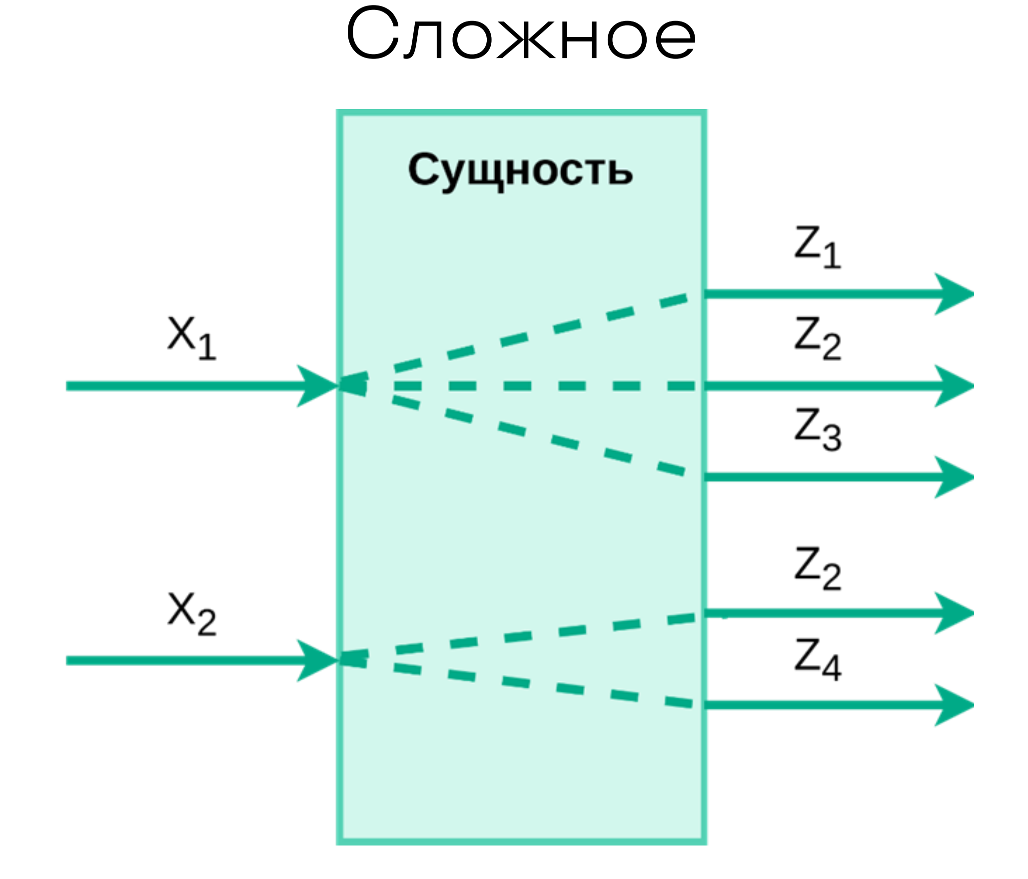
Программирование с явным выделением состояний





Поведение сущности









Состояние первично





Главный вопрос*:

В каких состояниях может находиться система?

*по мнению опрошенных экспертов





Состояние

- Несёт информацию о прошлом и настоящем





Состояние

- Несёт информацию о прошлом и настоящем
- Определяет реакцию на входное воздействие





Состояние

- Несёт информацию о прошлом и настоящем
- Определяет реакцию на входное воздействие
- Задаётся явно





Неявное состояние

```
if ($foo) {
    if ($bar || $bin) {
        if ($start) {
            // ...
        }
    } else {
        // ...
    }

    if ($chat) {
        // ...
    }
}

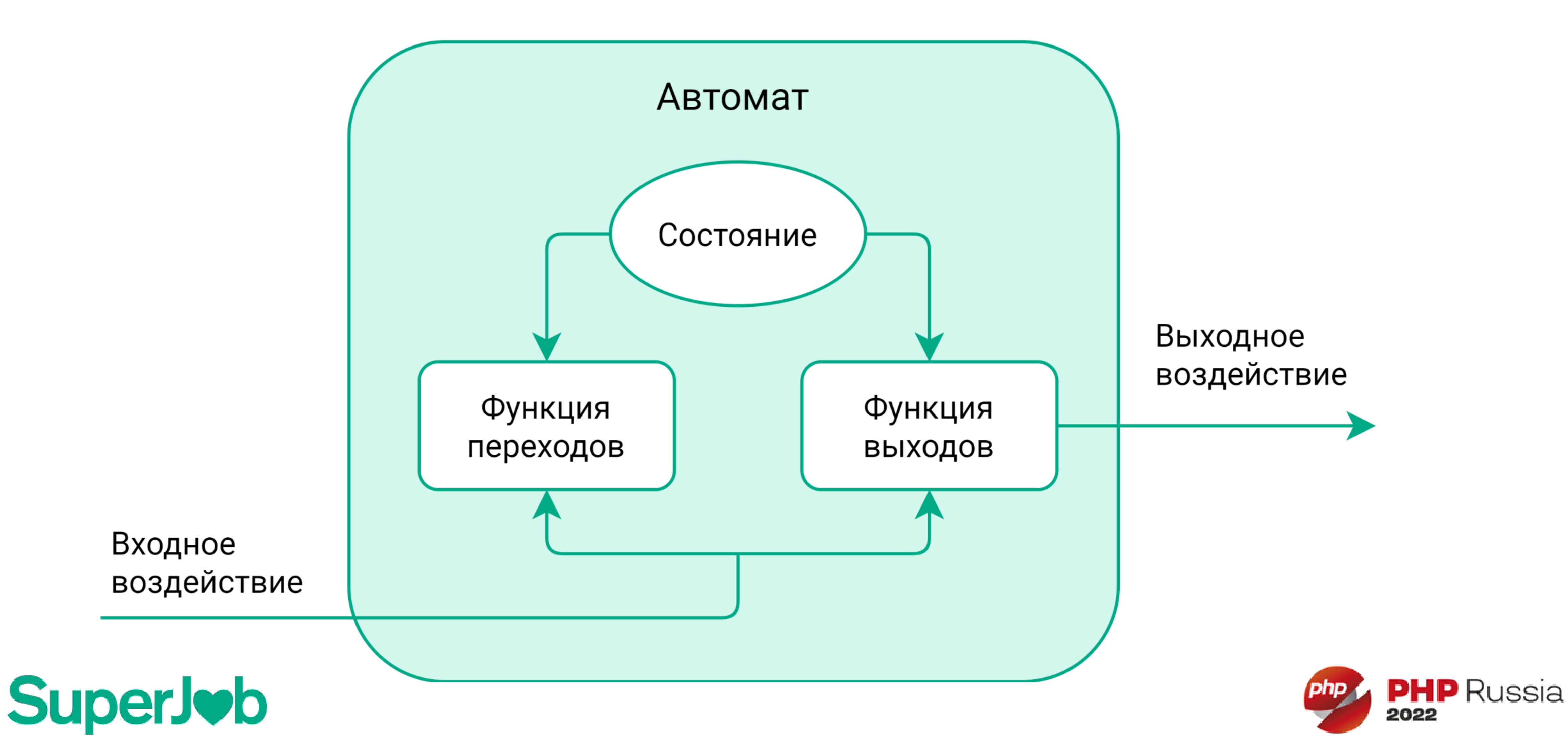
if ($bin) {
    // ...
}
```







Устройство автомата



Базовые требования

- Декларативное описание
- Простота расширения и модификации
- Ветвления
- Простота тестирования





Базовые требования

- Декларативное описание
- Простота расширения и модификации
- Ветвления
- Простота тестирования





Состояния в сценарии

1. starting
Отправка сообщения

2. await_answer
Ожидание ответа

Ответ

3. answer_received
Отправка сообщения





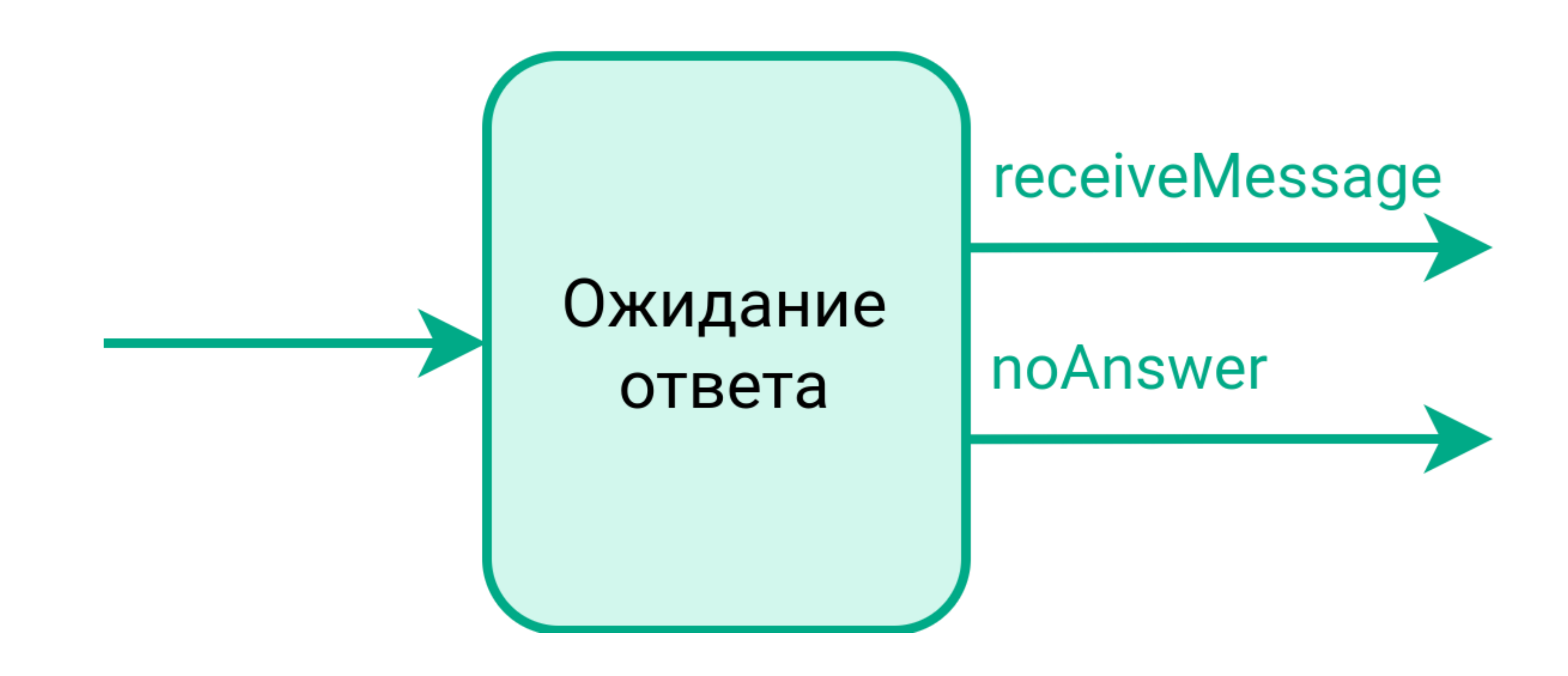
Описание состояний и переходов

```
[
  'starting' => [
    'actions' => [ /*...*/ ],
  ],
  'await_answer' => [
    'actions' => [ /*...*/ ],
  ],
  'answer_received' => [
    'actions' => [ /*...*/ ],
  ],
]
```





Действия



Структура действия

```
[
  'starting' => [ /*...*/ ],
  'await_answer' => [
    'actions' => [
      'receiveMessage' => [
        'next' => 'answer_received',
      ],
    ],
  ],
  'answer_received' => [ /*...*/ ],
]
```





Структура действия

```
[
  'starting' => [ /*...*/ ],
  'await_answer' => [
    'actions' => [
      'receiveMessage' => [
        'next' => 'answer_received',
      ],
    ],
  ],
  'answer_received' => [ /*...*/ ],
]
```





Структура действия

```
[
  'starting' => [ /*...*/ ],
  'await_answer' => [
    'actions' => [
      'receiveMessage' => [
        'next' => 'answer_received',
      ],
    ],
  ],
  'answer_received' => [ /*...*/ ],
]
```





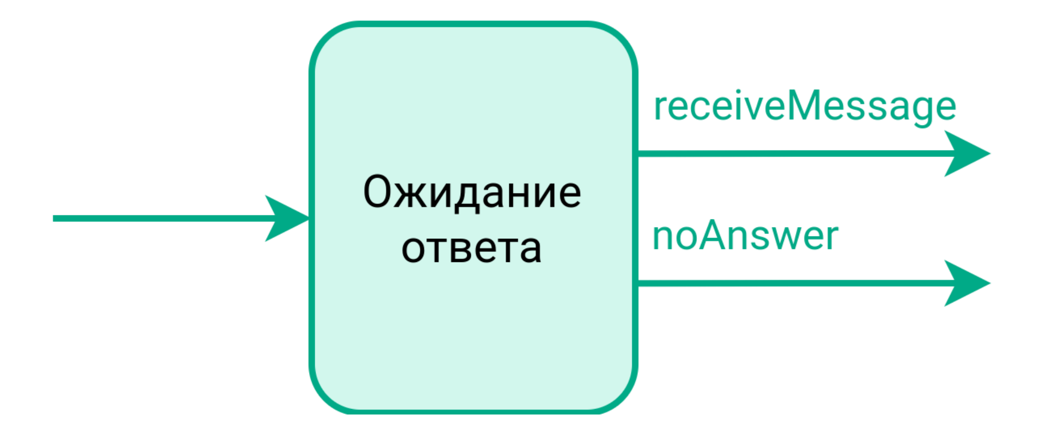
Управляющий объект

```
class Machine
{
  public function runAction(string $actionCode)
  {
    $action = $this->node->getAction($actionCode);

    if ($nextNodeId = $action->getNextNode()) {
        $this->node = $this->graph->transition($nextNodeId);
    }
  }
}
```

Запуск сценария

```
$scenario->runAction('receiveMessage');
```







Базовые требования

- ~~Декларативное описание~~
- Простота расширения и модификации
- Ветвления
- Простота тестирования





Executor

```
[
  'starting' => [ /*...*/ ],
  'await_answer' => [
    'actions' => [
      'receiveMessage' => [
        'executor' => MessageReceiver::class,
        'config' => [ /*...*/ ],
        'next' => 'answer_received',
      ],
    ],
  ],
  'answer_received' => [ /*...*/ ],
]
```





Executor

```
class MessageReceiver {
   public function run(Payload $payload, Config $config)
   {
      // выполнение логики

      return new Result(/* ... */);
   }
}
```





Особенности executor'a

- Независимый





Особенности executor'a

- Независимый
- Атомарный





Особенности executor'a

- Независимый
- Атомарный
- Простой





Управляющий объект

```
class Machine
{
  public function runAction(string $actionCode, Payload $payload)
  {
    $action = $this->node->getAction($actionCode);
    $result = $action->execute($payload);

    if ($nextNodeId = $action->getNextNode()) {
      $this->node = $this->graph->transition($nextNodeId);
    }
  }
}
```





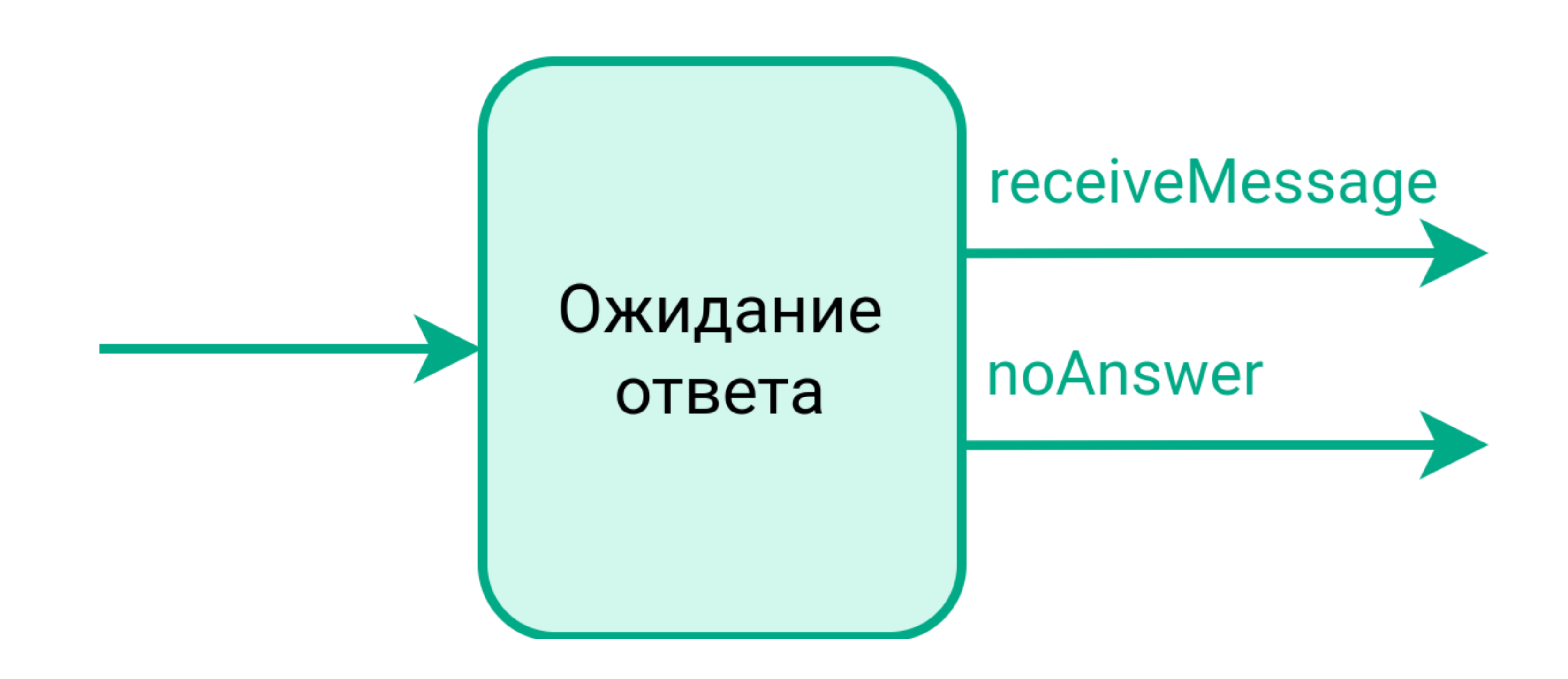
Базовые требования

- ~~Декларативное описание~~
- ~~Простота расширения и модификации~~
- Ветвления
- Простота тестирования





Ветвления



Ветвления

Да

receiveMessage

Ожидание ответа

Нет

noAnswer





Описание схемы ветвления

```
'await_answer' => [
  'actions' => [
    'receiveMessage' => [
      'executor' => MessageReceiver::class,
      'next' => [
        'да' => 'answer_received',
        'нет' => 'end'
      ],
    ],
  ],
]
```





Ветвления в управляющем объекте

```
class Machine
{
  public function runAction(string $actionCode, Payload $payload)
  {
    $action = $this->node->getAction($actionCode);
    $result = $action->execute($payload);

    if ($nextNodeId = $action->getNextNode($result)) {
      $this->node = $this->graph->transition($nextNodeId);
    }

    return $result;
  }
}
```





Автоматические переходы

Да

Отправка сообщения

sendMessage

Ожидание ответа

receiveMessage

Ожидание ответа

noAnswer

Нет

Автоматические переходы

Ожидание ответа

receiveMessage

noAnswer

Да

Нет

Отправка сообщения

sendMessage

Ожидание ответа

Условия автоматического перехода

- Есть только одно действие
- Тип узла «automaitic»





Автоматические переходы

```
class Machine
{
  public function run(string $actionCode, Payload $payload)
  {
    do {
      $result = $this->runAction($actionCode, $payload);
      $payload = new Payload($result);
    } while ($result && $this->node->isAutomatic());
  }
}
```





Выделение ветвлений в отдельный узел

Ожидание ответа

receiveMessage

noAnswer

Да

Нет

Не знаю





Базовые требования

- ~~Декларативное описание~~
- ~~Простота расширения и модификации~~
- ~~Ветвления~~
- Простота тестирования





Простота тестирования

- Тестируем executor'ы отдельно





Простота тестирования

- Тестируем executor'ы отдельно
- Тестируем основные кейсы сценария отдельно



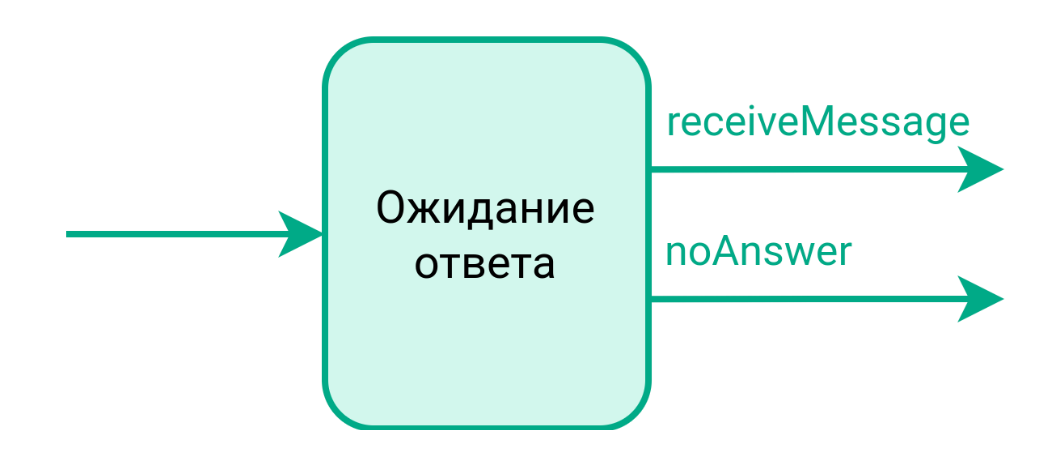


Простота тестирования

- Тестируем executor'ы отдельно
- Тестируем основные кейсы сценария отдельно
- Тестируем только ожидаемые действия

Только ожидаемые действия

- receiveMessage
- noAnswer
- goAway

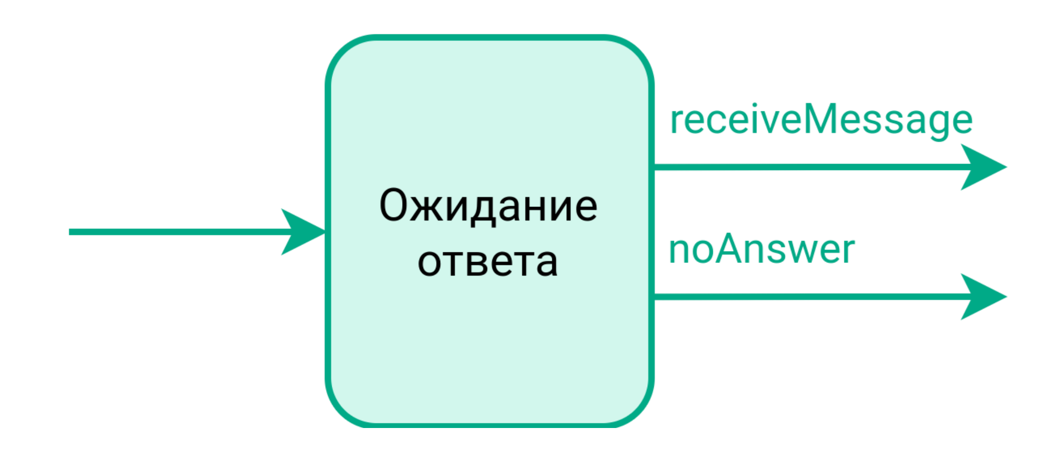






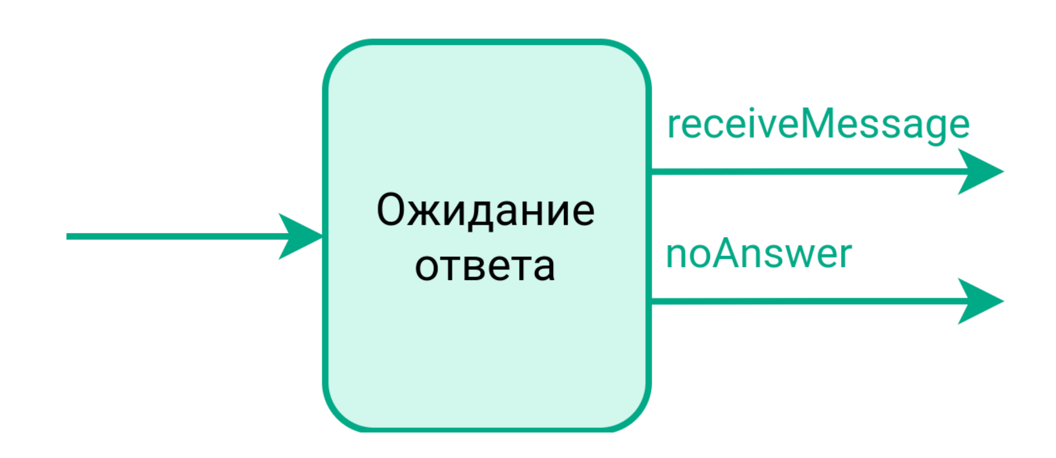
Только ожидаемые действия

- receiveMessage
- noAnswer
- goAway



Только ожидаемые действия

- receiveMessage
- noAnswer
- ~~goAway~~







Базовые требования

- ~~Декларативное описание~~
- ~~Простота расширения и модификации~~
- ~~Ветвления~~
- ~~Простота тестирования~~

Улучшаем фреймворк

- Хранилище данных
- Версионирование
- Управление восстановлением
- Аналитика
- Проблема роста сценария





Улучшаем фреймворк

- Хранилище данных
- Версионирование
- Управление восстановлением
- Аналитика
- Проблема роста сценария



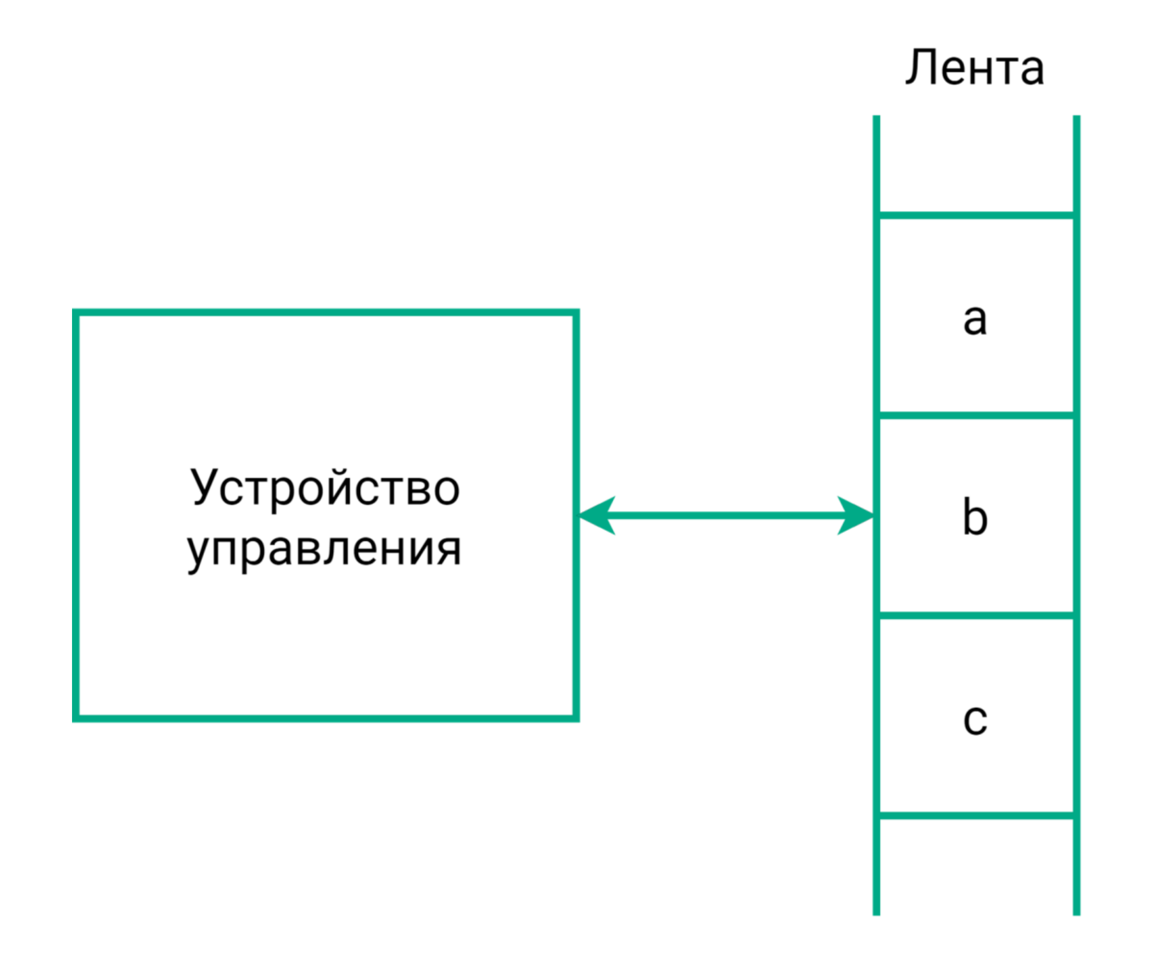


Где хранить информацию для сценария?





Машина Тьюринга







Лента = Хранилище





Разница состояний

- Состояние управляющего объекта – действия
- Хранилище – результат этих действий





Улучшаем фреймворк

- ~~Хранилище данных~~
- Версионирование
- Управление восстановлением
- Аналитика
- Проблема роста сценария





Версионирование

V1

V2

V3





Улучшаем фреймворк

- ~~Хранилище данных~~
- ~~Версионирование~~
- Управление восстановлением
- Аналитика
- Проблема роста сценария





Управление восстановлением

Сохраняем:

- Входные параметры
- Состояние
- Хранилище

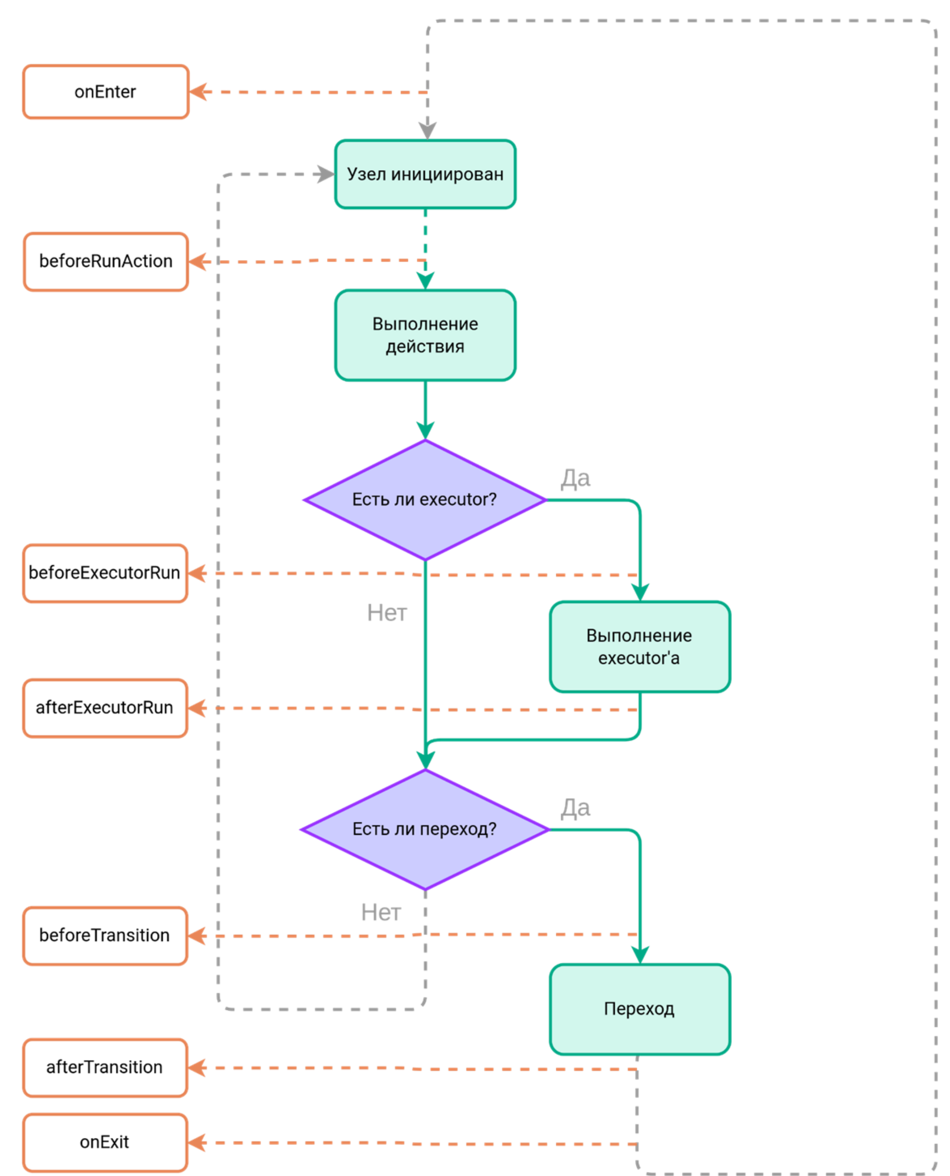
Улучшаем фреймворк

- ~~Хранилище данных~~
- ~~Версионирование~~
- ~~Управление восстановлением~~
- Аналитика
- Проблема роста сценария





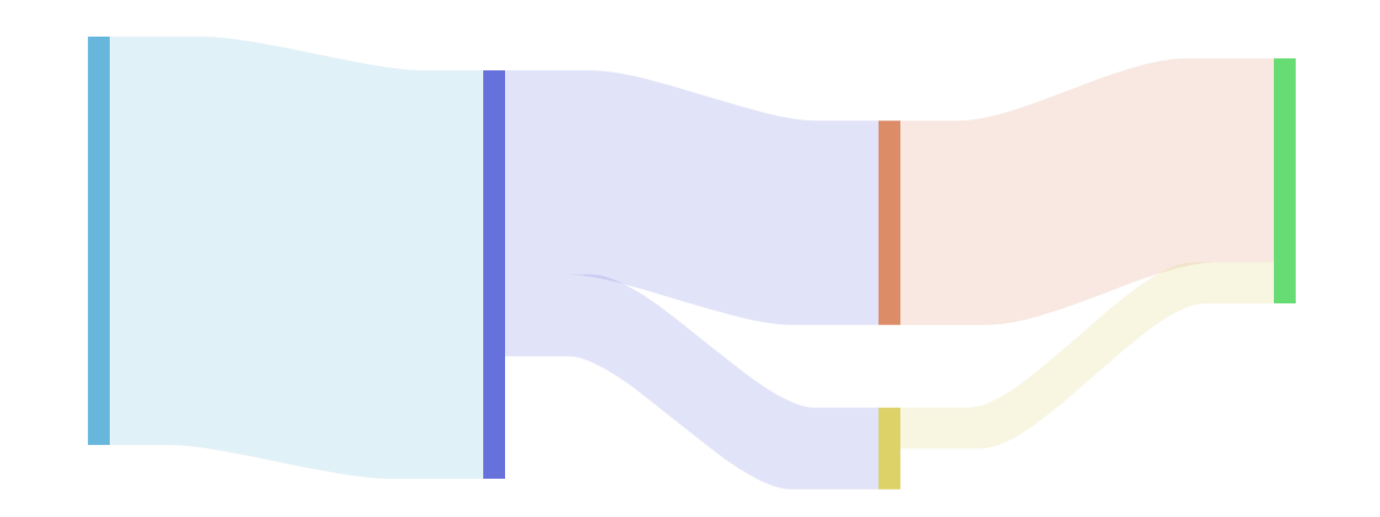
Жизненный цикл

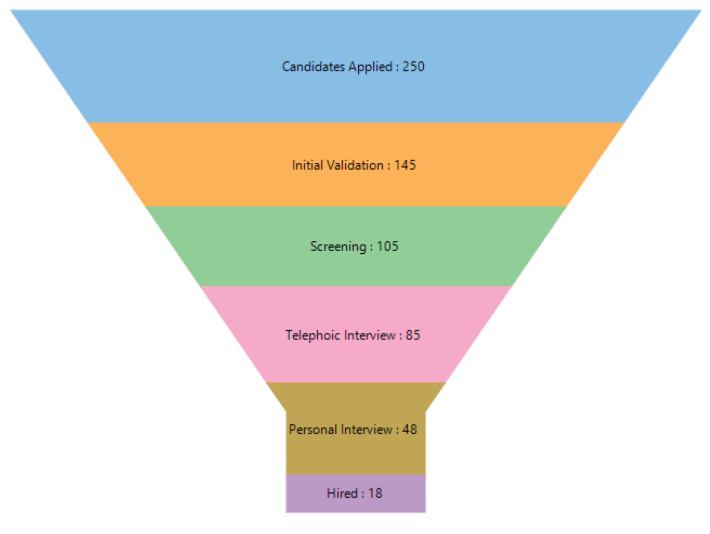






Аналитика




Candidates Applied : 250
Initial Validation : 145
Screening : 105
Telephoic Interview : 85
Personal Interview : 48
Hired : 18
Conversion rate for successful hiring is 7.2%


SuperJob


php
PHP Russia
2022

Улучшаем фреймворк

- ~~Хранилище данных~~
- ~~Версионирование~~
- ~~Управление восстановлением~~
- ~~Аналитика~~
- Проблема роста сценария





Проблема роста сценария

- Слишком много узлов
- Дублирование логики





Иерархические автоматы





Подавтомат: уточнение ответа

Да

Отправка сообщения

Ожидание ответа

условие

Нет

default

Уточнение ответа





Соблюдаем правила иерархии

- Внутренние узлы не могут делать переходы наружу
- Внешние узлы не могут делать переходы внутрь
- Хранилища не пересекаются





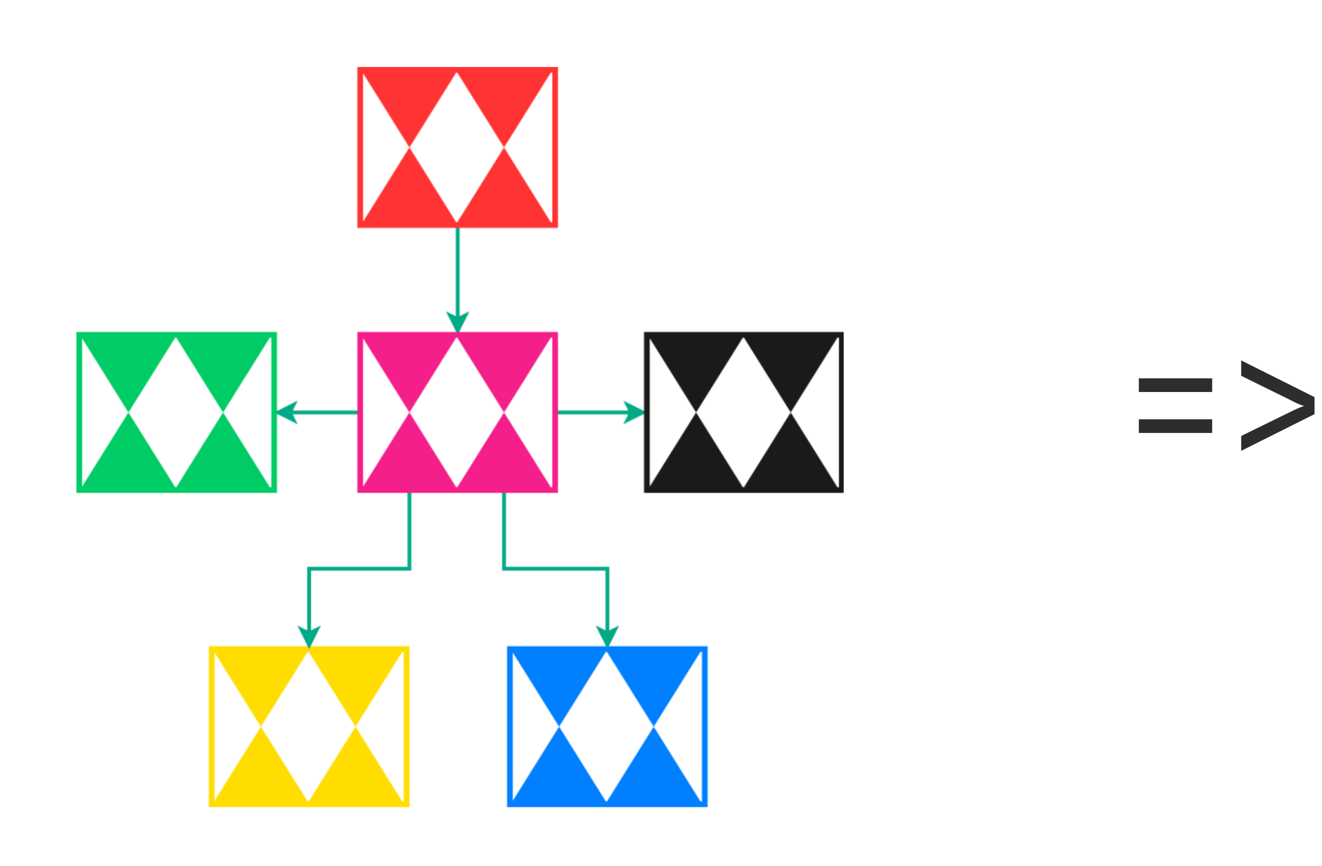
Структура подавтомата

```
'poll' => [
  'actions' => [ /* … */ ],
  'states' => [
    // Состояния подавтомата
    'sub_state_1' => [ /* … */ ],
    'sub_state_2' => [ /* … */ ],
  ],
]
```





Могучие подавтоматы. Мегабот



=>



Схема стала проще

SuperJob



Результаты работы

- 4 основных сценария, 3 на подходе
- Более 20 одноразовых сценариев
- 21 итерация одного из сценариев
- Минимальное время запуска сценария — 1 час

Аналоги

- BPMS и т.п.
- Symfony Workflow
- Temporal





Аналоги

- BPMS и т.п.
- Symfony Workflow
- Temporal
- Сотни других решений



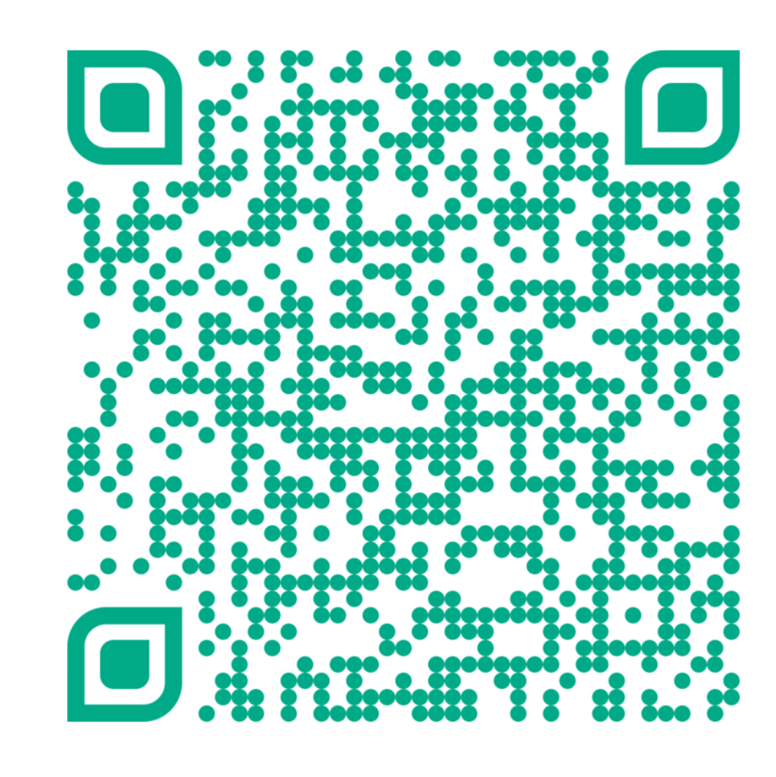


Спасибо за внимание!

Telegram: @gavrilov_evgeny

Полезные ссылки:

https://tinyurl.com/phprussia-ap



Оцените доклад



